# REMOTE COMPUTING USING THE NATIONAL FUSION GRID

J.R. Burruss[1,*], S. Flanagan[1], K. Keahey[2], C. Ludescher[3], D.C. McCune[3], Q. Peng[1], L. Randerson[3], D.P. Schissel[1], M. Thompson[4]

[1]General Atomics, P.O. Box 85608, San Diego, California 92186-5608

[2]Argonne National Lab, 9700 S. Cass Avenue, Argonne, Illinois 60439

[3]Princeton University, Princeton, New Jersey 08543

[4]Lawrence Livermore National Laboratory, P.O. Box 808, Livermore, California 94551

* *corresponding author, email burruss@fusion.gat.com, phone (858) 455-2865*

**Abstract**

The National Fusion Collaboratory (http://www.fusiongrid.org) uses grid technology to implement remote computing on the National Fusion Grid. The motivations are to reduce the cost of computing resources, shorten the software deployment cycle, and simplify remote computing for the user community.

The National Fusion Collaboratory has successfully demonstrated remote access as a grid service to the TRANSP transport analysis code for tokamak experiments. TRANSP development and administration are now centralized at the Princeton Plasma Physics Laboratory (PPPL), obviating both the need to port TRANSP to different platforms and the process of deploying TRANSP to remote sites. TRANSP users now share the resources of a powerful Linux cluster located at PPPL. Fusion researchers have completed over 900 TRANSP runs utilizing over 5,600 hours of CPU time since the TRANSP service was installed in October 2002.

KEYWORDS: grid computing, TRANSP, MDSplus, National Fusion Grid

## 1. Introduction to Grid Computing

In a traditional computing environment software users typically install and run programs on a local machine. This requires that developers create and maintain versions of their software for the different platforms used by their users. This also requires users to update their local installations as the software is updated.

In a grid computing environment applications, systems, and other computing resources are abstracted into services [1]. This abstraction allows users to invoke services on local or remote hosts without concerning themselves with the details of how such services are implemented. Just as users of the power grid need not learn the details of which power plants produce power when they plug an appliance into an outlet, so too can computational grid users ignore the details of where their computing power comes from when they use a grid service.

Computational grid users must be able to invoke services that may run on remote computers. Therefore any computational grid must include the capability to invoke codes remotely, and the interface for remote invocation should be as uniform as possible.

Transparent access to the resources of a computational grid is facilitated through the use of a single sign-on mechanism; users need not enter a username and password for each host of a grid, but instead sign on to the entire grid. In other words, users login into the grid, not into individual hosts.

Grid computing can also reduce the cost of computing resources by facilitating resource sharing. It is much less expensive for users of a grid to share a single Linux cluster than for each user to purchase their own cluster, for example.

## 2. The National Fusion Grid

The National Fusion Collaboratory solved the challenges of grid-wide authentication and remote invocation through the use of the Globus Toolkit [2]. The Globus Toolkit is a set of programs and protocols that solve the classic grid problems of resource allocation and authentication. For example, to run a code remotely one must allocate resources on the remote system or systems. The Globus Resource Allocation Manager (GRAM) simplifies the task of allocating resources on remote computers. Rather than "reinvent the wheel" application programmers can leverage GRAM when writing codes that allocate resources on remote systems. Authentication is another common problem in remote computing. Programmers developing grid applications using the Globus Toolkit have access to the Globus Grid Security Infrastructure (GSI). X.509 certificates—digital documents that map a public key to a user name and are digitally signed by a Certification Authority (CA) to assure authenticity—are used by GSI as grid-wide authentication tokens to provide users with a single sign-on. This allows all applications to use the same authentication scheme—different applications on different systems can be constructed to be instantly interoperable as far as authentication is concerned.

The need for grid-wide authentication means that grid administrators must agree on at least one CA that all grid participants acknowledge as a trusted authority for issuing X.509 identities. The CA for the National Fusion Grid is the DOE Grids CA, https://pki1.doegrids.org/.

Authorization is handled using the Akenti security engine [3]. Akenti uses X.509 certificates, allows for very detailed resource usage requirements, and supports the kind of distributed access control essential for a grid environment.

Resource monitoring is handled by the Fusion Grid Monitor (FGM). The FGM is used to monitor the status of services as well as the status of individual code runs. Users access the FGM through a web browser. The FGM is discussed in detail in [4].

## 3. Grid-enabled TRANSP

TRANSP is a transport analysis code for tokamak experiments [5]. On the National Fusion Grid TRANSP is available as a grid service. TRANSP is currently run on a cluster of Linux machines at PPPL. TRANSP users can invoke TRANSP on this cluster of Linux machines directly from PPPL computers, or remotely from hosts on the National Fusion Grid. An example of running TRANSP through the National Fusion Grid is given in Fig. 1.

The installation of TRANSP as a grid service has had a very positive impact. To illustrate, take the example of TRANSP users at DIII-D. For DIII-D users, each individual TRANSP run executes 4-5 times faster than the old locally-installed TRANSP. Also, DIII-D users can now run several TRANSP runs in parallel. Because administration is centralized at PPPL, DIII-D programmers no longer need to spend time maintaining a local copy of TRANSP; this is a significant savings because DIII-D programmers used to spend 3 programmer months a year maintaining a local copy of TRANSP. Another benefit is that the most up-to-date version of TRANSP is always available to DIII-D researchers. Researchers used to need to wait for programmers to install and test TRANSP updates—this deployment cycle has been eliminated by making TRANSP available as a grid service.

To run grid-enabled TRANSP through the National Fusion Grid, users must first load their inputs into an MDSplus database. MDSplus is a data acquisition and storage system popular in the magnetic fusion community [6]. As a database MDSplus is hierarchical; data are organized into trees that contain a root node and many child nodes. Each node in an MDSplus tree may have multiple child nodes. MDSplus trees may be accessed through native file I/O, or remotely though a client/server connection over TCP/IP. Both the inputs and outputs of TRANSP are stored in MDSplus trees. If a client site has an MDSplus server installation then the process of creating and loading the MDSplus tree can be done locally; if there is no local MDSplus installation, an MDSplus server is available at the PPPL server site. This step of creating a tree and loading inputs can be done by hand, through the use of PPPL-provided shell scripts, or by using the IDL-based PreTRANSP application.

Regardless of how inputs are loaded, this newly created TRANSP tree must be accessible through a Globus-enabled MDSplus server so that TRANSP Grid services can access the TRANSP inputs stored in the tree. When the TRANSP run eventually completes, outputs will be written to this same MDSplus tree. Sites without their own MDSplus server can also receive TRANSP output in the traditional NetCDF[1] file format via GridFTP[2]; a PPPL-provided script makes this GridFTP task very simple.

Before using any grid services, users must sign on to the National Fusion Grid by creating a temporary proxy certificate. This proxy is created using the Globus Toolkit routine grid-proxy-init. Users can only generate a valid proxy if they already have a valid X.509 certificate. The X.509 certificates are valid for one year, after which they must be renewed. Proxies typically have a very limited lifespan; the default lifespan is 12 hours. Although this is long enough for most TRANSP runs to complete, users are advised to give their proxies a lifespan of 10 days to allow runs to be restarted at PPPL under an unexpired proxy in case of technical difficulties at the server site or with TRANSP itself, which is an evolving research code. Alternatively, a user can extend the lifetime of the proxy when they realize that it might expire during the computation. The proxy certificate is used by TRANSP Grid services; the services will take this proxy certificate and make connections using the proxy on behalf of the user. If the proxy were

[1] NetCDF (Network Common Data Form) is a self-describing platform independent file format
[2] GridFTP is a file transfer program provided by the Globus Toolkit that transfers data—possibly in parallel—over a secure grid connection

to expire before the TRANSP run completed, then the results could not be written back to the originating MDSplus server, so it is important to select a proxy lifespan of sufficient duration.

To use an analogy, a proxy certificate is to an X.509 certificate what a travel visa is to a passport. A passport is a form of identification issued by a trusted authority. Likewise, an X.509 certificate identifies a user and is issued by a trusted Certification Authority. In contrast, a travel visa is a temporary document used on a single trip, and a proxy certificate is a temporary document typically used for a single code run.

It is important to note that users executing a code run on the PPPL Linux cluster need not deal with accounts on the PPPL Linux cluster computers. When a collaborator runs TRANSP on the National Fusion Grid, connections made by their proxy are mapped to PPPL-assigned "run production" accounts created specially for the TRANSP service. These run production accounts are implemented as local UNIX accounts on the PPPL cluster, and are used to ensure data privacy. Users never need to learn a new set of passwords or host names as this account mapping happens behind the scenes. This greatly simplifies the task of account administration.

When users request TRANSP Grid services, their proxy is used to verify their identity through the Globus GSI. Once authenticated, users are authorized to use services via the Akenti policy engine. Akenti is used by TRANSP administrators to specify levels of access for different classes of TRANSP user. The Akenti database identifies users though the use of the same X.509 certificates used for authentication. Typically TRANSP users are mapped to a "Clients" group and are authorized to run only those services used by TRANSP. Other groups include the "General" group which permits certain test services to be invoked, and "Developers" which permits access to all services.

Users tell the TRANSP Grid service where to find the MDSplus TRANSP tree for a run using .REQUEST files; these .REQUEST files are small text files containing a list of different variables, and are generated by PreTRANSP or the PPPL scripts. File input through a .REQUEST file is used instead of the standard input stream (stdin) to work around the problem of firewalls at client sites that restrict input traffic. The .REQUEST files are sent to the PPPL computers using Globus routine globus-url-copy. After the .REQUEST file is sent, users invoke the Globus routine globus-job-submit to request that TRANSP be run on the PPPL computers on their behalf. Both of these commands communicate with the Globus gatekeeper service on transpgrid.pppl.gov. These interactions are automated by PreTRANSP or the PPPL scripts.

After globus-job-submit is used to start the TRANSP run on the National Fusion Grid, PPPL computers connect to the MDSplus server specified in the .REQUEST file, read TRANSP inputs from the specified TRANSP tree, and start execution of the TRANSP code on the PPPL Linux cluster. The PPPL computers send status messages to the Fusion Grid Monitor (FGM) so that users can monitor the progress of the TRANSP runs remotely. Run logfiles are made available through the PPPL TRANSP website (http://w3.pppl.gov/transp/log) or through the FGM. After TRANSP runs complete, the TRANSP Grid service again connects to the originating MDSplus server on the user's behalf using the user's proxy certificate, and writes the TRANSP outputs to the same tree that contains the TRANSP inputs. Alternatively, a NetCDF file containing the final TRANSP outputs can be returned using GridFTP. A final status message is sent to the Fusion Grid Monitor, and an email is sent to the user communicating that the TRANSP run is complete.

The process of preparing TRANSP inputs, managing code runs, and launching TRANSP is made simpler through the use of the IDL-based PreTRANSP application [7]. PreTRANSP was first used to dispatch TRANSP runs onto the National Fusion Grid in 2002. For now at least, PreTRANSP is only used for the preparation of TRANSP runs. However, the basic pattern of a data-driven GUI that manages code runs, loads inputs into MDSplus, and dispatches codes may be repeated for future grid services. (Figure 1)

**4. New Developments and Future Work**

The next grid service to be added to the National Fusion Grid is GS2. GS2 is a physics code used to study low-frequency turbulence in magnetic plasmas [8]. A prototype of this new grid-enabled GS2 runs on a Beowulf cluster at the University of Maryland. Users already have the ability to invoke GS2 remotely using the standard Globus interface from the command line; eventually a web-based portal will provide users with a more convenient interface.

The host gk.umd.edu serves as a Globus gateway for the prototype grid-enabled GS2. Users prepare GS2 inputs in an ASCII text file, and then submit the input file to the Globus gatekeeper on gk.umd.edu. The Globus software then submits the GS2 job for execution on the Beowulf cluster on behalf of the user. When the job completes, results are sent back to gk.umd.edu. The results are then sent directly back to the user in an ASCII format table.

Ultimately, the command line interface used to invoke GS2 will be replaced with a web-based portal. This web interface will likely be implemented using a Java Applet, but the details of this have not been decided upon.

Other plans for GS2 include storing complete calculation information in MDSplus, and getting more sophisticated inputs from MDSplus (consisting mainly of kinetic profile information from TRANSP and equilibrium information from TRANSP and EFIT).

Additionally monitoring capabilities will be added by leveraging the existing Fusion Grid Monitor.


**Acknowledgment**

Work supported by U.S. Department of Energy under DOE Grant DE-FC02-01ER25455.

**References**


[1] I. Foster, C. Kesselman, J. Nick, S. Tuecke, The Physiology of the Grid: An Open Grid Services Architecture for Distributed Systems Integration, Open Grid Service Infrastructure WG, Global Grid Forum, June 22, 2002.

[2] I. Foster, C. Kesselman, Globus: A Metacomputing Infrastructure Toolkit, International Journal of Supercomputing Applications, 11(2):115-128, 1997.

[3] M. Thomson, A. Essiari, S. Mudumbai, Certificate-based Authorization Policy in a PKI Environment, ACM Transactions on Information and System Security, August 2003, http://www-itg.lbl.gov/Akenti

[4] S. Flanagan, J. R. Burruss, C. Ludescher, D.C. McCune, Q. Peng, L. Randerson, D.P. Schissel, A general-purpose data analysis system with case studies from the national fusion grid and the DIII-D MDSplus between pulse analysis system, Fusion Engineering and Design, July 2003.

[5] TRANSP, http://w3.pppl.gov/transp/

[6] MDSplus, http://www.mdsplus.org

[7] J. Burruss, Using PreTRANSP at DIII-D, http://web.gat.com/comp/analysis/grid/pretransp.html

[8] GS2, http://gs2.sourceforge.net

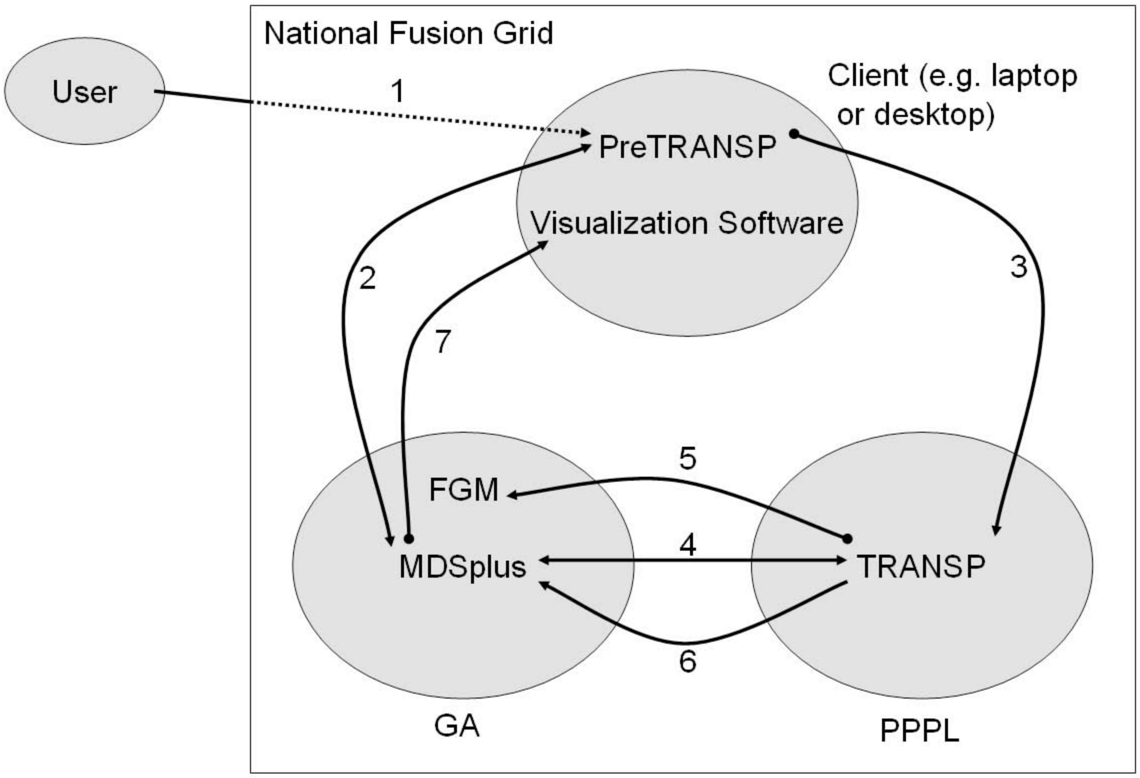


Figure 1: Example TRANSP run on the National Fusion Grid. 1) User uses grid-proxy-init to sign on to the grid, 2) PreTRANSP loads inputs into MDSplus, 3) PreTRANSP requests TRANSP run at PPPL, 4) TRANSP loads inputs from MDSplus, 5) During execution, status messages are posted to the FGM, 6) TRANSP outputs are written to MDSplus, 7) Visualization software used to view TRANSP outputs.